CI Report (Gender Detection)

Members :

Ashkan Mousazade

Mahshid Alizade

Hadise Khalili

University Of Guilan

Professor : Mr.Tourani

1399.Tir

مقدمه

به طور معمول، انسان می‌تواند به راحتی بین مرد و زن در تصاویر تمایز قائل شود، اما تشریح دقیق اینکه چرا می‌تواند چنین کاری را به این سادگی انجام دهد، دشوار است. بدون تعیین ویژگی‌های دقیق، این تمایز می‌تواند برای رویکردهای یادگیری ماشین سنتی خیلی دشوار باشد. علاوه بر آن، ویژگی‌هایی که مرتبط با انجام این کار هستند همیشه دقیقا به یک شیوه بیان نمی‌شوند، بلکه برای هر انسانی کمی متفاوت به نظر می‌رسد. الگوریتم‌های یادگیری عمیق، راهکاری برای پردازش اطلاعات بدون ویژگی‌های از پیش تعریف شده ارائه می‌کنند و پیش‌بینی‌های دقیقی را با وجود تنوع در چگونگی بیان ویژگی‌ها انجام می‌دهند

پیاده سازی

۱. کتابخانه ها:

ابتدا کتابخانه هایی که برای پیاده سازی این پروژه به آن نیاز داریم را import می کنیم :

```
import cv2 as cv
import argparse
```

- **پکیج CV2 از کتابخانه ی OpenCV :**

OpenCV یا همان **O**pen **C**omputer **V**ision **L**ibrary مجموعه ای از کتابخانه‌های برنامه‌نویسی پردازش تصویر و یادگیری ماشین است که مجموعه ای از توابع از پیش آموزش دیده را برای تشخیص چهره در خود دارد که میتوان به سادگی از آن استفاده کرد.

- **کتابخانه ی argparse :**

از این کتابخانه برای خواندن از cmd و کار با آن استفاده می کنیم

۲. شناسایی تصویر :

با استفاده از تابع getFaceBox چهره هایی که در تصویر مشاهده میکنیم را شناسایی میکنیم :

```
def getFaceBox(net, frame, conf_threshold=0.7):
    frameOpencvDnn = frame.copy()
    frameHeight = frameOpencvDnn.shape[0]
    frameWidth = frameOpencvDnn.shape[1]
    blob = cv.dnn.blobFromImage(frameOpencvDnn, 1.0, (300, 300), [104, 117, 123], True, False)

    net.setInput(blob)
    detections = net.forward()
    bboxes = []
    for i in range(detections.shape[2]):
        confidence = detections[0, 0, i, 2]
        if confidence > conf_threshold:
            x1 = int(detections[0, 0, i, 3] * frameWidth)
            y1 = int(detections[0, 0, i, 4] * frameHeight)
            x2 = int(detections[0, 0, i, 5] * frameWidth)
            y2 = int(detections[0, 0, i, 6] * frameHeight)
            bboxes.append([x1, y1, x2, y2])
            cv.rectangle(frameOpencvDnn, (x1, y1), (x2, y2), (0, 255, 0), int(round(frameHeight / 150)), 8)
    return frameOpencvDnn, bboxes
```

در این تابع به ترتیب زیر عمل میکنیم :

- ابتدا کپی ای از تصویر ورودی تهیه میکنیم و طول و عرض آن را در دو متغیر ذخیره میکنیم
- با استفاده از cv.dnn.blobFrameImage تصویر را به فرم دلخواه با سایز مورد نظر در می آوریم ، openCv این تابع را برای تسهیل پردازش تصویر برای طبقه بندی یادگیری عمیق فراهم می کند.
- آن را به اطلاعات قبلی اضافه میکنیم
- با forward. همه ی تصاویر شناسایی شده را در detection ذخیره می کنیم.
- و در حلقه ی for ابعاد آن ها را استخراج میکند با کمک cv.rectangle دور چهره های شناسایی شده یک مستطیل رسم میکنیم

توضیحات مربوط به این خط در تابع بالا :

```
blob = cv.dnn.blobFromImage(frameOpencvDnn, 1.0, (300, 300), [104, 117, 123], True, False)
```

- **image** :

این تصویر ورودی است که می خواهیم قبل از انتقال آن از طریق شبکه عصبی عمیق ما برای طبقه بندی، پردازش کنیم . دارای ۴ ویژگی زیر است.

- **scale factor** :

پس از انجام تفریق متوسط ، می توانیم بصورت اختیاری تصاویر خود را براساس فاکتور مقیاس بندی کنیم. این مقدار به مقدار ۱.۰ (به عنوان مثال، بدون مقیاس) پیش فرض است اما می توانیم مقدار دیگری نیز ارائه دهیم .

- **size** :

در اینجا ما اندازه فضایی را که از شبکه عصبی Convolutional انتظار دارد ارائه می دهیم. برای اکثر شبکه های عصبی فعلی این حالت ۲۲۴ × ۲۲۴ ، ۲۲۷ × ۲۲۷ یا ۲۹۹ × ۲۹۹ است .

- **mean** :

اینها مقادیر تفریق ما هستند. آنها می توانند یک سه ضلعی از RGB باشند یا می توانند یک مقدار واحد باشند که در این صورت مقدار عرضه شده از هر کانال تصویر کم میشود. اگر تفریق متوسط را انجام می دهید، اطمینان حاصل کنید که ترتیب ۳ تاپل را در (R ، G، B)مرتب باشد. به خصوص هنگام استفاده از رفتار پیش فرض swapRB = True

- **swapRB** :

openCV فرض می کند که تصاویر به ترتیب کانال BGR هستند. با این حال، میانگین مقدار فرض می کند که ما از دستور RGB استفاده می کنیم. برای برطرف کردن این اختلاف می توانیم با تنظیم این مقدار در True ، کانال های R و B را در تصویر جابجا کنیم. به طور پیش فرض، OpenCV این تعویض کانال را برای ما انجام می دهد .

- **crop** :

flag ی ست که مشخص میکند آیا تصویر بعد از resize کراپ خواهد شد یا نه .

- **ddepth** :

عمق blobِ خروجی ست . (int)

۳. تنظیمات مربوط به cmd :

در این قسمت ، تنظیمات مربوط به کارکردن با cmd را اعمال میکنیم :

```
parser = argparse.ArgumentParser()
parser.add_argument("-i", help='Give image file')
```

در این دو خط کد ، ابتدا یک آبجکت از argparse میسازیم و ورودی به cmd را بررسی میکنیم که اگر در ورودی i- آمده بود در ادامه ی آن باید آدرس فایلی را که میخواهیم آن را پردازش کنیم ببینیم .

پس در خط دوم آدرس فایل ورودی ، جهت شناسایی چهره و تعیین جنسیت ، را میدهیم .

در cmd ورودی را به صورت زیر میدهیم :

- ابتدا مسیر cmd را به جایی که فایل py. قرار دارد میبریم
- سپس مینویسیم : python [.py name].py -i [picture name].jpg

۴. معرفی مدل ها و وزن ها :

شناسایی جنسیت با استفاده از امکان های نمایش OpenCV بسیار مشهور است و برخی از شما ممکن است در مورد آن نیز کار کرده یا خوانده باشید . در بسته dnn ، OpenCV کلاسی به نام Net ارائه داده است که می تواند برای جمع آوری یک شبکه عصبی مورد استفاده قرار گیرد. علاوه بر این، این بسته ها از وارد کردن مدل های شبکه عصبی از چارچوب های یادگیری عمیق شناخته شده مانند caffe ، tensorflow پشتیبانی می کنند. محققان مدل های CNN خود را به عنوان مدل کافه (caffe Model) منتشر کرده اند.

یک caffe model دارای دو فایل به هم مرتبط است :

۱. prototxt :

تعریف cnn در اینجا آمده است.

این پرونده لایه های شبکه عصبی، ورودی ها، خروجی ها و عملکرد هر لایه را تعریف میکند.

۲. caffe model :

این شامل اطلاعات شبکه ی عصبی آموز دیده (مدل آموزش دیده) است .

که آن ها را در این قسمت معرفی میکنیم :

```
faceProto = "opencv_face_detector.pbtxt"
faceModel = "opencv_face_detector_uint8.pb"

genderProto = "gender_deploy.prototxt"
genderModel = "gender_net.caffemodel"
```

و در بخش بعدی Net را از مدل ها استخراج میکنیم :

```
genderNet = cv.dnn.readNetFromCaffe(genderProto, genderModel)
faceNet = cv.dnn.readNet(faceModel, faceProto)
```

۵.لیست خروجی :

لیست خروجی های احتمالی را در این بخش مینویسم که میتواند Female یا Male باشد :

```
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
genderList = ['Male', 'Female']
```

۶.پردازش ویدیو :

در ادامه برای اینکه بتوانیم علاوه بر عکس ، ویدیو هم پردازش کنیم :

```
cap = cv.VideoCapture(args.i if args.i else 0)
```

و از متغیر cap فریم ها را استخراج میکنیم .

۷. تشخیص جنسیت :

```
while cv.waitKey(1) < 0:
    # Read frame
    hasFrame, frame = cap.read()
    if not hasFrame:
        cv.waitKey()
        break
    frameFace, bboxes = getFaceBox(faceNet, frame)
    if not bboxes:
        print("No face Detected ")
        continue

    for bbox in bboxes:
        # print(bbox)
        face = frame[max(0, bbox[1] - padding):min(bbox[3] + padding, frame.shape[0] - 1),
               max(0, bbox[0] - padding):min(bbox[2] + padding, frame.shape[1] - 1)]

        blob = cv.dnn.blobFromImage(face, 1.0, (227, 227), MODEL_MEAN_VALUES, swapRB=False)
        genderNet.setInput(blob)
        genderPreds = genderNet.forward()
        gender = genderList[genderPreds[0].argmax()]

        print("Gender : {}, confidence = {:.3f}".format(gender, genderPreds[0].max()))

        label = "{}".format(gender)
        cv.putText(frameFace, label, (bbox[0] - 5, bbox[1] - 10), cv.FONT_HERSHEY_SIMPLEX, 0.55, (0, 255, 0), 1,cv.LINE_AA)
        cv.imshow("Gender ", frameFace)
```

در این بخش در حلقه ی while تا زمانی که کلیدی زده نشده تصویر پردازش میشود.

در ابتدا چهره های شناسایی شده استخراج میشوند سپس بررسی میشود که اگر frame ی وجود نداشت از برنامه خارج میشود و اگر چهره ای تشخیص داده نشد ارور میدهد و در حلقه ی for :

- مشخصات چهره ی شناسایی شده در متغیر face ذخیره میشود
- در blob به فرمی که برای پردازش راحت است ذخیره میشود
- و بعنوان ورودی به **genderNet** فرستاده میشود و مقدار ورودی جدید برای شبکه را تنظیم می کند
- با متد **forward**. جنسیت تمام چهره های شناسایی شده ، تشخیص داده میشود و در **genderPreds** ذخیره میشوند ، سپس جنسیت آن فرد مورد نظر که قرار از در صفحه نمایش داده شود در **gender** نوشته میشود
- حالا بر اساس اینکه جنسیت تشخیص داده شده چه است ، در بالای کادر نوشته میشود و تصویر به نمایش در می آید.

مثالی از نحوه ی ورودی دادن و خروجی گرفتن (عکس و ویدیو) :

ورودی :

همانطور که گفته شد ، در cmd ورودی را به صورت زیر میدهیم :

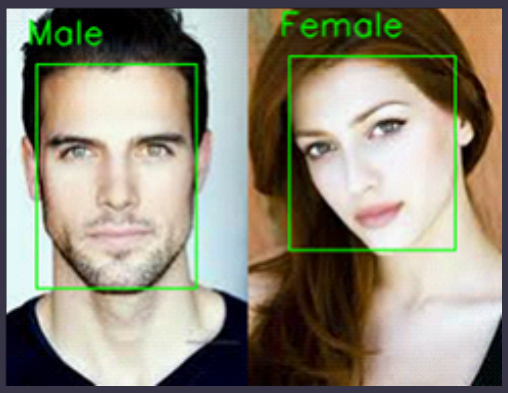
- ابتدا مسیر cmd را به جایی که فایل py. قرار دارد میبریم
- سپس مینویسیم : python [file name].py -i [picture name].jpg

```
C:\Users\user>cd desktop

C:\Users\user\Desktop>cd gender-detection

C:\Users\user\Desktop\gender-detection>python gender.py -i 1.jpg
Gender : Male, confidence = 1.000
Gender : Female, confidence = 0.987
```

و خروجی : (در cmd میتوان درصد اعتماد از حدسی که زده شده را مشاهده کرد)




```
C:\Users\user\Desktop\gender-detection>python gender.py -i 1.jpg
Gender : Male, confidence = 1.000
Gender : Female, confidence = 0.987
```

استفاده از WebCam بعنوان ورودی :

در ادامه قصد داریم علامه بر عکس و ویدیو ، از WebCam هم ورودی بگیریم .

فایل py. دیگری به نام gender_RealTime ، در فایل پروژه ی ارسالی وجود دارد که با اعمال تغییرات کمی در آنچه تا حالا توضیح دادیم بدست آمده است و از آن برای پردازش ورودی وب کم استفاده میکنیم .

کتابخانه های جدید :

کتابخانه ی جدیدی که اضافه می شود imutils نام دارد که به کمک آن از videoStream پشتیبانی میکنیم .

- برای استفاده از این کتابخانه ، لازم است ابتدا آن را نصب کنید (از cmd کمک میگیریم) :

```
C:\Users\user>pip install imutils
```

- سپس آن هارا بصورت زیر import میکنیم :

```
from imutils.video import VideoStream
import argparse
import imutils
import cv2
```

- قبل از ورود به حلقه ی while ، وب کم را روشن میکنیم و پیغامی مبنی بر روشن شدن ان چاپ میکنیم:

```
print("[INFO] starting video stream...")
vs = VideoStream(src=0).start()
```

- در ادامه ، بجای اینکه تصویر را از فایل ورودی که به برنامه میدادیم بخوانیم ، از آبجکتی که از **videoStream** ساختیم می خوانیم :

```
# Read frame
frame = vs.read()
frameFace, bboxes = getFaceBox(faceNet, frame)
```

- حالا باید کلیدی را برای بسته شدن و ختمه دادن به وب کم تعریف کنیم ، که در اینجا می گوییم به محض فشرده شدن کلید Q وب کم بسته و برنامه خاتمه داده شود **(توجه کنید زبان کیبورد انگلیسی باشد)** :

```
    key = cv2.waitKey(1) & 0xFF

    if key == ord("q"):
        break

cv2.destroyAllWindows()
vs.stop()
```

نحوه ی ورودی دادن و خروجی گرفتن(WebCam) :

ورودی :
در اینجا هم مثل حالت قبل از cmd کمک میگیریم با این تفاوت که دیگر لزومی به وارد کردن مسیر فایل ورودی نداریم :

- ابتدا مسیر cmd را به جایی که فایل py. قرار دارد میبریم
- سپس مینویسیم : python [file name].py

```
C:\Users\user>cd desktop

C:\Users\user\Desktop>cd gender-detection

C:\Users\user\Desktop\gender-detection>python gender_realTime.py
```

و خروجی :

وب کم باز میشود و در صورتی که چهره ای شناسایی کند دور آن مربع میکشد و جنسیت را مینویسد .
اما اگر چهره ای در کادر نباشد اررورِ "No face Detected " می دهد.

نکات مهم پایانی :

۱. قابلیت های **جدیدی** که به پروژه ی خواسته شده اضافه کردیم :

- قابلیتِ دادن ویدیو به ورودی (علاوه بر عکس)
- استفاده از وب کم و تعیین جنسیت بصورت RealTime

۲. سایت های مفیدی که به فهم کد خیلی کمک میکنند :

- کتابخانه یArgparse
- مثال هایی از کارکردن با OpenCV
- مدل های tensorflow و متد های ReadNet